IN THE CLAIMS

1. through 15. (Canceled)

16. (Currently Amended) A computer-implemented method for measuring a complexity of nested object state transition diagrams that are represented as data structures in the computer, the method comprising:

a) using [a] the computer, determining a plurality of graphs of object state transitions at K levels l_k, wherein:

(A) $0<k<K$;

(B) one or more of said graphs at level l_k+1 are expansions of one or more of said graphs at level l_k; and

(C) said graphs comprise a plurality of nodes to represent a corresponding plurality of states of use-cases and a plurality of edges to represent a corresponding plurality of transitions between the states; and

b) using the computer, determining a complexity for said plurality of graphs.

17. (Previously Presented) The computer-implemented method of claim 16, wherein:

the state of an object at level k is a super state of an object state at level k+1, and

the state of an object at level k is a sub state of an object state at level k-1, and

the method further includes identifying the state transion relationship between super state and sub state as one of a group consisting of "in," "out," and "inout".

18. (Previously Presented) The computer-implemented method of claim 17, further comprising:

selecting said level $l_k = l_K$;

expanding a graph L from said selected level l_k with at least one graph from said level l_{K+1};

determining paths in said expanded graph; and

determining a number of conditions in said transition paths.

19. (Previously Presented) The computer-implemented method of claim 18 further comprising:

removing an unnecessary path from said at least one graph.

20. (Previously Presented) The computer-implemented method of claim 19 wherein said one or more unnecessary paths comprise one or more numbers of a group consisting of:

(exception → fallout),

(exception → exception),

(one-repetition loop →fallout),

(one repetition loop → cancelled), and

(one repetition loop → exception).

21. (Previously Presented) ~~A computer-implemented method of claim 16, to~~ A computer-implemented method for measuring a complexity of nested object state transition diagrams that are represented as data structures in the computer, the complexity measuring method comprising:

a) using the computer, determining a plurality of graphs of object state transitions at K levels l_k, wherein:

(A) $0<k<K$;

(B) one or more of said graphs at level l_k+1 are expansions of one or more of said graphs at level l_k; and

(C) said graphs comprise a plurality of nodes to represent a corresponding plurality of states of use-cases and a plurality of edges to represent a corresponding plurality of transitions between the states; and

b) using the computer, determining a complexity for said plurality of graphs, the complexity determining step including measuring [e] a nested object state transition complexity between two super-states by recursively applying Equation 6:

$$STPC_{k,p,q} = \sum_{i=1}^{m_k} \left(\prod_{j=1}^{n_{k,i}} (C_{k,i,j} + (STPC_{k+1,i,j} - sub_{k,i,j})) + mul_{k,i}(N) \right) \qquad \text{Equation 6}$$

wherein:

m_k = a number of transition paths of level-k object (k>1) between two super-states or a number of transition paths of level-1 object;

$n_{k,i}$= a number of states along path i for level-k object;

$C_{k,i,j}$ is a number of conditions to bring level-k object state from Sk,i,j-1 to Sk,i,j ;

$STPC_{k+1,i,j}$ = Substate Transition Complexity between state Sk,i,j-1 and Sk,i,j;

$sub_{k,i,j}$=0 if $STPC_{k+1,i,j}$=0 and $sub_{k,i,j}$=1 if ($STPC_{k+1,i,j}\neq 0$) ;

if a multiplicity between level-k object and $level_{-(k+1)}$ is 1:N, then $mul_{k,i}(N)=0$ if (N=1) and $mul_{k,i}(N)=r$ if (N>1); and

r is a number of times that $level_{-k+1}$ objects transit back to $level_{-k}$ in path i.

22. (Canceled)

Add the following new claims 23-26:

23. (New) The computer-implemented method of claim 21, wherein:

the state of an object at level k is a super state of an object state at level k+1, and

the state of an object at level k is a sub state of an object state at level k-1, and

the method further includes identifying the state transion relationship between super state and sub state as one of a group consisting of "in," "out," and "inout".

24. (New) The computer-implemented method of claim 23, further comprising:

selecting said level $l_k = l_K$;

expanding a graph L from said selected level l_k with at least one graph from said level l_{K+1};

determining paths in said expanded graph; and

determining a number of conditions in said transition paths.

25. (New) The computer-implemented method of claim 24 further comprising:

removing an unnecessary path from said at least one graph.

26. (New) The computer-implemented method of claim 25 wherein said one or more unnecessary paths comprise one or more numbers of a group consisting of:

(exception → fallout),

(exception → exception),

(one-repetition loop →fallout),

(one repetition loop → cancelled), and

(one repetition loop → exception).